CLAIMS

What is claimed is:

1.   A method of creating point-in-time view of data on a disk, comprising:

initiating from a host, a first session of writing data to a disk which
5   affects a portion of the disk;

creating and storing entries in an array on the disk which identify where
the data written to the disk during said first session is located;

initiating at least one second session of writing data to a disk at a time
different from initiation of the first session, with said at least one second session of
10   writing data affecting a portion of the disk;

copying data in any portion of the disk corresponding to said first
session which is to be affected by a write operation by said at least one second
session;

creating and storing entries in said array which identify where data
15   written to the disk during said session is located; and

invalidating said entries in said array for said at least one of said first
and said second session when at least one of a new first session and a new second
session is initiated.

20   2.   The method of claim 1, wherein a session sequence number is assigned to a
session when a session is initiated, and said sequence number is stored in said array.

3.   The method of claim 2, wherein as input and output operations are processed,
the session sequence number is stored in a chunk allocation block.

25

4.   The method of claim 1, wherein said data affected by said at least one second
session write operation is copied from a source disk to be stored at a cache disk.

5.   The method of claim 1, wherein said first session and said at least one second
30   session are initiated and controlled by a first host, and point-in-time viewing of the
data on the disk is conducted by a second host.

6.   The method of claim 1, wherein sessions are assigned specific slots in the array, and wherein when a new session is to replace a corresponding prior session, it is assigned to the same slot in the array as the prior session.

5   7.   The method of claim 6, wherein said sessions are assigned session ID's, and wherein the session ID of a prior invalidated session is different from the session ID of a later corresponding session assigned to its same slot in the array.

8.   The method of claim 1, wherein said array comprises a direct linear map
10   pointing to sections of the disk where data is located for each session.

9.   The method of claim 8, wherein said direct linear map comprises a Map Region subsystem, a Paging subsystem, a Chunk Repository subsystem and a Direct Linear Map subsystem.
15

10.   The method of claim 1, wherein each session is assigned a unique session ID.

11.   A method of allowing point-in-time view of data on a disk, for data written to a disk throughout a plurality of different sessions, comprising
20   creating an array on a disk comprised of a map which stores entries which point to locations on a disk where data for different sessions is located;
               assigning predetermined slots for corresponding sessions in said array; and
               assigning unique session ID for each session for which entries are
25   stored in said array.

12.   The method of claim 11, further comprising invalidating a session by assigning a new session corresponding thereto to the invalidated session's slot in the array, and assigning the new session an ID different from that of the invalidated session.
30

13.   The method of claim 11, wherein said array comprises a direct linear map pointing to sections of the disk where data is located for each session.

14.   The method of claim 13, wherein said direct linear map comprises a Map Region subsystem, a Paging subsystem, a Chunk Repository subsystem and a Direct Linear Map subsystem.

5   15.   The method of claim 14, wherein said VM region subsystem presents all VM metadata as a set of 64KB VM regions.

16.   The method of claim 14, wherein said paging subsystem maintains a least recently used number of pages so that unused pages can be used to read in new VM

10 regions.

17.   The method of claim 14, wherein the Chunk Repository subsystem manages the cache disk as a set of mappable Chunks.

15 18.   The method of claim 14, wherein the Direct Linear Map subsystem maintains a map from the source LU offset, session and LU write bit to a mappable Chunk.